L Number	Hits	Search Text	DB	Time stamp
-	513	((merg$3 or combin$9) with sort$3 with (list or criteria or condition or instruction)) and (@ad<20001006)	USPAT; US-PGPUB; IBM_TDB	2004/09/01 21:01
-	133	((merg$3 or combin$9) with sort$3 with (list or criteria or condition or instruction)) and (@ad<20001006) and 707/$.ccls.	USPAT; US-PGPUB; IBM_TDB	2004/02/21 07:53
-	102	((merg$3 or combin$9) with sort$3 with (list or criteria or condition or instruction)) and (implie$5 or imply$3) and (@ad<20001006)	USPAT; US-PGPUB; IBM_TDB	2004/02/20 22:26
-	24	((merg$3 or combin$9) with sort$3 with (list or criteria or condition or instruction)) and ((implie$5 or imply$3) with (data or information or field or element)) and (@ad<20001006)	USPAT; US-PGPUB; IBM_TDB	2004/09/01 20:20
-	1	"20020078023"	USPAT; US-PGPUB; IBM_TDB	2004/02/21 07:25
-	18	(city or cities) with (impli$3 or imply$3) and (@ad<20001006)	USPAT; US-PGPUB; IBM_TDB	2004/02/21 07:26
-	25	(city or cities) with (impli$3 or imply$3 or express$3) and (@ad<20001006) and 707/$.ccls.	USPAT; US-PGPUB; IBM_TDB	2004/02/21 07:42
-	503	((implie$5 or imply$3) with (data or information or field or element)) and (@ad<20001006) and 707/$.ccls.	USPAT; US-PGPUB; IBM_TDB	2004/09/01 14:08
-	4	(correlat$3 with sort$3 with (list or criteria or condition or instruction)) and (@ad<20001006) and 707/$.ccls.	USPAT; US-PGPUB; IBM_TDB	2004/02/21 07:50
-	134	((merg$3 or combin$9 or correlat$3) with sort$3 with (list or criteria or condition or instruction)) and (@ad<20001006) and 707/$.ccls.	USPAT; US-PGPUB; IBM_TDB	2004/02/21 07:53
-	1	"20020087517"	USPAT; US-PGPUB; IBM_TDB	2004/09/01 11:07
-	1	((sort$5 adj (parameter or field or element or data)) same ((implie$5 or imply$3) with (additional or "another" or "other" or relat$7 or associat$5) with (parameter or field or element or data))) and (@ad<20001006)	USPAT; US-PGPUB; IBM_TDB	2004/09/02 10:59
-	3	4701840.pn. or 4468732.pn. or 5018060.pn.	USPAT; US-PGPUB; IBM_TDB	2004/09/01 14:13
-	2	(4701840.pn. or 4468732.pn. or 5018060.pn.) and sort$3 and impl$5	USPAT; US-PGPUB; IBM_TDB	2004/09/01 14:15
-	1	(sort$3 same (default with (last adj name))) and (@ad<20001006)	USPAT; US-PGPUB; IBM_TDB	2004/09/01 20:21
-	3	((merg$3 or combin$9 or add$7) with (default near2 sort$3) with (user near3 sort$3)) and (@ad<20001006)	USPAT; US-PGPUB; IBM_TDB	2004/09/02 16:36
-	0	(sort$4 adj parameter) near2 ((impli$5 or imply$5) same addition$3 same parameter) and (@ad<20001006)	USPAT; US-PGPUB; IBM_TDB	2004/09/02 11:01
-	0	((sort$4 adj (parameter or term or identifier or element)) near2 (impli$5 or imply$5)) and (@ad<20001006)	USPAT; US-PGPUB; IBM_TDB	2004/09/02 11:03
-	45	((sort$4 adj (parameter or term or identifier or element)) same (impli$5 or imply$5 or associat$3 or link$3)) and (@ad<20001006) and (database or table)	USPAT; US-PGPUB; IBM_TDB	2004/09/02 11:05
-	76	((sort$4 adj (parameter or term or identifier or element)) same (impli$5 or imply$5 or associat$3 or link$3)) and (@ad<20001006)	USPAT; US-PGPUB; IBM_TDB	2004/09/02 11:09

-	18	((sort$4 adj (parameter or term or identifier or element)) same (impli$5 or imply$5 or associat$3 or link$3)) and (@ad<20001006) and (707/$.ccls. or 705/$.ccls. or 715/$.ccls.)	USPAT; US-PGPUB; IBM_TDB	2004/09/02 14:57

```
Set     Items   Description
S1    2698518   SORT? OR ARRANG? OR INDEX? OR ORGANISE OR ORGANISING? OR O-
             RGANIZ?
S2    5945227   FIRST OR 1ST OR INITIAL OR PRIMARY
S3    4068088   2ND? OR SECOND? OR ADDITIONAL? OR BROADER?
S4   12745043   CRITERI? OR SPECIFIC? OR PROPERT? OR FEATUR? OR IDENTIFIER?
              OR ELEMENT?
S5   11655439   MERGE? OR MERGING OR RELAT? OR LINK? OR SUBCATEGOR? OR ASS-
             OCIAT?
S6    1000441   REPEAT? OR RESORT OR RE()SORT? OR AGAIN OR ITERAT?
S7     919306   DATABASE? OR DATABANK? OR DATA()(BASE? OR BANK?) OR DB OR -
             OODB OR RDB OR DBMS OR RDBMS
S8         54   S1 AND S2 AND S3 AND S4 AND S5 AND S6  AND S7
S9      22887   S1(2N)(S2 OR S3)
S10         3   S9 AND S4 AND S5 AND S6 AND S7
S11        56   S8 OR S10
S12        52   RD (unique items)
S13        44   S12 NOT PY>2000
S14        44   S13 NOT PD=20001006:20031006
S15        44   S14 NOT PD=20031006:20040901
S16        44   S15 NOT PC>20001006
S17       197   S9 AND S4 AND S5 AND S7
S18    420882   TIER? OR HIERARCH? OR NESTED OR MULTILEVEL? OR SUBLEVEL?
S19        16   S17 AND S18
S20        11   RD (unique items)
S21        11   S20 NOT S11
S22         8   S21 AND S2
S23         6   S22 NOT PY>2000
File   8:Ei Compendex(R) 1970-2004/Aug W2
         (c) 2004 Elsevier Eng. Info. Inc.
File  35:Dissertation Abs Online 1861-2004/Jul
         (c) 2004 ProQuest Info&Learning
File 202:Info. Sci. & Tech. Abs. 1966-2004/Jul 12
         (c) 2004 EBSCO Publishing
File  65:Inside Conferences 1993-2004/Aug W3
         (c) 2004 BLDSC all rts. reserv.
File   2:INSPEC 1969-2004/Aug W2
         (c) 2004 Institution of Electrical Engineers
File  94:JICST-EPlus 1985-2004/Jul W4
         (c)2004 Japan Science and Tech Corp(JST)
File 111:TGG Natl.Newspaper Index(SM) 1979-2004/Aug 16
         (c) 2004 The Gale Group
File 233:Internet & Personal Comp. Abs. 1981-2003/Sep
         (c) 2003 EBSCO Pub.
File   6:NTIS 1964-2004/Aug W3
         (c) 2004 NTIS, Intl Cpyrght All Rights Res
File 144:Pascal 1973-2004/Aug W2
         (c) 2004 INIST/CNRS
File  34:SciSearch(R) Cited Ref Sci 1990-2004/Aug W3
         (c) 2004 Inst for Sci Info
File  62:SPIN(R) 1975-2004/Jun W3
         (c) 2004 American Institute of Physics
File  99:Wilson Appl. Sci & Tech Abs 1983-2004/Jul
         (c) 2004 The HW Wilson Co.
```

Set	Items	Description
S1	3717804	SORT? OR ARRANG? OR INDEX? OR ORGANI?
S2	2441499	FIRST OR 1ST OR INITIAL OR PRIMARY
S3	2483940	2ND? OR SECOND? OR ADDITIONAL? OR BROADER?
S4	3650212	CRITERI? OR SPECIFIC? OR PROPERT? OR FEATUR? OR IDENTIFIER? OR ELEMENT?
S5	2086582	RELAT? OR LINK? OR SUBCATEGOR? OR ASSOCIAT?
S6	20858	S1 AND S2 AND S3 AND S4 AND S5
S7	503199	REPEAT? OR RESORT OR RE()SORT? OR AGAIN OR ITERAT?
S8	1507	S6 AND S7
S9	75549	S1(5N)S3
S10	175	S8 AND S9
S11	15	S10 AND IC=G06F?
S12	13	S10 AND (DATABASE? OR DB OR DATA()(BASE? OR BANK?) OR OODB-?)
S13	7	S12 NOT S11
S14	16	S8 AND IC=G06F-007?
S15	13	S14 NOT S11
S16	20	S15 OR S13
S17	18	S16 NOT ORGANISM?
S18	29051	HIERARCH? OR TIER? OR SUBCATEGOR? OR SUBLEVEL?
S19	6426	S1 AND S18
S20	280	S7 AND S19
S21	12	S20 AND IC=G06F-007?
S22	32	S20 AND (DATABASE? OR DATABANK? OR DATA()(BASE? OR BANK?) - OR DB OR OODB OR RDB)
S23	42	S21 OR S22
S24	35	S11 OR S12 OR S13 OR S16
S25	42	S23 NOT S24
S26	39	S25 NOT ORGANISM?
S27	32	S26 NOT AD>20001006
S28	32	IDPAT (sorted in duplicate/non-duplicate order)
S29	32	IDPAT (primary/non-duplicate records only)

File 347:JAPIO Nov 1976-2004/Apr(Updated 040802)

File 350:Derwent WPIX 1963-2004/UD,UM &UP=200453

Sorting

WILLIAM A. MARTIN*

Massachusetts Institute of Technology, Cambridge, Massachusetts 02139

The bibliography appearing at the end of this article lists 37 sorting algorithms and 100 books and papers on sorting published in the last 20 years. The basic ideas presented here have been abstracted from this body of work, and the best algorithms known are given as examples. As the algorithms are explained, references to related algorithms and mathematical or experimental analyses are given. Suggestions are then made for choosing the algorithm best suited to a given situation.

Key words and phrases: sorting

CR category: 5.31

INTRODUCTION

Sorting is used to put items in order. The sorting algorithms themselves are not difficult to understand, but a comparison of the relative merits of the many algorithms does require some effort. In fact, the question of when an ordering is required is not a simple one: for example, a file that is best maintained in sorted order when stored in magnetic tape might be kept more efficiently on disk with a scatter storage technique. Whether or not a file should be sorted depends on how it is to be used, the extent to which the storage medium can be randomly accessed, and the statistics of the particular item of information on which the file might be sorted. Once the various sorting algorithms have been analyzed, one can see how these factors come into play.

In data management applications it is customary to define a *file* as a collection of records, and a *record* as consisting of one or more *information groups.* Each information group may contain several *items of information.* Records within a file are often sorted, and sometimes information groups and individual items of information are sorted as well. Records are sorted by identifying a particular item of information in the record as the *key;* the records are then sorted into the ascending or descending order of their keys. Most sorting schemes involve moving the elements to be sorted from one place to another. The elements are generally moved several times before the final sorted order is achieved. Thus, when sorting records it may be better either to sort their keys first and then move the records into the final sorted order, or to use the sorted keys as an index to the records. Questions of this type will be considered in the latter sections of this paper, after presentation of the various sorting algorithms.

No one sorting technique is best for every situation. The fastest methods are more difficult to program and are not considered worth the effort for a few short lists of numbers. Even if programming effort is not a consideration, the choice of method would depend on: the length of the list to be sorted; the relation between the length of the list and the number of cells in the main memory of the machine used for sorting; the number and size of any disk or tape units used in the sort; the extent to which the elements are

* The work herein was supported in part by Project MAC, an MIT research project sponsored by the Advanced Research Projects Agency, Department of Defense, under Office of Naval Research Contract N00014-70-A-0362-0001.

The Use of Information in Sorting

H. LYNN BEUS

General Electric Company, Schenectady, New York*

ABSTRACT. The information-gathering aspect of sorting is considered from a theoretical viewpoint. A large class, R, of sorting algorithms is defined, based on the idea of information use. Properties of this algorithm class are developed, and it is noted that several well-known sorting algorithms are closely related to algorithms in R. The Binary Tree Sort is shown to be in R and to have unique properties in this class. A vector is defined which characterizes the information-gathering efficiency of the algorithms of R. Finally, a more general class of algorithms is defined, and some of the definitions extended to this class. Two intriguing conjectures are given which appear to require graph theory or combinatorial topology for their solution.

KEY WORDS AND PHRASES: sorting, sorting efficiency, sorting theory, sorting algorithms, information use, graph theory, combinatorial analysis, binary tree

CR CATEGORIES: 5.30, 5.31, 5.32, 5.6

1. *Introduction and Basic Definitions*

In the last twenty years the introduction of digital computers has spurred interest in the problems of sorting or ordering sets of data. The stream of papers that began with Mauchly [8] and Goldstine and Von Neumann [5] have, in the main, presented results of individual algorithms for sorting, rather than general results concerning the overall sorting problem. There have been several exceptions to this, notable among which are [2, 3].

In this paper we restrict ourselves to the study of one aspect of the sorting process in an attempt to discover some unifying results which will apply to a great many algorithms. It has been pointed out that sorting can be considered as consisting of two intermixed processes: information gathering and ordering. In the first the items are compared in some way to gain information about their relative order. In the second the information is used to carry out the actual ordering of the items. It is the first of these processes that we study here in an attempt to obtain some measure of the efficiency of various algorithms in information-gathering.

In this discussion we call the set of items to be sorted a *sort set*. Herein such sets are arbitrary in general, but fixed for each particular application. We use x to denote such a standard sort set. Then x is a set of n items to be sorted, and if a, b are two items of x, πa and πb denote the values of their keys (on which the sorting is being done). We assume that items of x are numbered $1, 2, \cdots, n$, and that the key values also are integers from the set $\{1, 2, \cdots, n\}$. I_n denotes the set of integers $\{1, 2, 3, \cdots, n\}$, and $|G|$ the cardinality of a set G.

Given a sort set x to be sorted by an algorithm A, there are $n!$ possible sequences of key values associated with the items of x. We define the efficiency, ωA, of the

This paper is a revision of the author's Ph.D. thesis (Computing Center Rep. No. 1096) at Case Western Reserve University, Cleveland, Ohio, Nov. 1967.
* Research and Development Center.